PTO 05-5671

CY=WO DATE=20000331 KIND=A1
PN=00-60460

GENERIC PROCESS FOR ASSISTING THE MAPPING OF SIGNAL PROCESSING APPLICATIONS ONTO PARALLEL COMPUTERS

[Procede generique d'aide au placement d'applications de traitement de signal sur calculateurs paralleles]

Juliette Mattioli, et al.

UNITED STATES PATENT AND TRADEMARK OFFICE
Washington, D.C August 2005

Translated by: FLS, Inc.

PUBLICATION COUNTRY	(19)	WO
DOCUMENT NUMBER	(11)	00/60460
DOCUMENT KIND	(12)	A1
PUBLICATION DATE	(43)	20001012
APPLICATION NUMBER	(21)	PCT/FR00/00824
DATE OF FILING	(22)	20000331
ADDITION TO	(61)	
INTERNATIONAL CLASSIFICATION	(51)	G06F 9/46, 17/50
PRIORITY	(30)	99/04182
INVENTORS	(72)	MATTIOLI, JULIETTE GUETTIER, CHRISTOPHE JOURDAN, JEAN
APPLICANT	(71)	THOMSON CSF
DESIGNATED CONTRACTING STATES	(81)	AE AG AL AM AT AU AZ BR BY CA CH CN CR CU CZ DE DK DM DZ EE ES FI GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MA MD MG MK MN MW MX NO NZ PL PT RO RU SD SE SG SI SK SL TJ TM TR TT TZ UA UG US UZ VN YU ZA ZW ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).
TITLE	(54)	GENERIC PROCESS FOR ASSISTING THE MAPPING OF SIGNAL PROCESSING APPLICATIONS ONTO PARALLEL COMPUTERS
FOREIGN TITLE	[54A]	Procede generique d'aide au placement d'applications de traitement de signal sur calculateurs paralleles

/1*

This invention relates to a generic process for assisting the mapping of signal processing applications onto parallel computers.

Mapping consists in distributing data and operations associated with processing, such as a signal processing application, onto a computer, generally a computer with parallel architecture. This mapping is static, since all mapping choices are made before execution of the mapped application, in contrast to dynamic mapping.

Numerous software tools and environments are known for performing this mapping. Among others, there are the SynDEx tool from INRIA for signal and image processing, the PTOLEMY tool from the University of Berkeley, HPF for scientific computation, the "FX compiler", GEDEA from Lockheed Martin,... However few known tools allow complete automation of mapping. Moreover, even if the general objective of the tools such as SynDEx or GEDEA is to provide the user assistance in the development and optimization of real time applications implemented on a multiprocessor architecture which is generally heterogeneous, for purposes of quickly implementing prototypes, these tools manage only one crude level of granularity (granularity being the degree of fineness and precision which is to be obtained for an application and its implementation on a given architecture). The Ptolemy environment is essentially an environment for simulating and prototyping heterogenous systems integrating hardware and software.

Moreover, all these known systems generally allow evaluation of the performance of mapping for a given solution, while indicating the most efficient communication network between the processors, as well as

---

* Number in the margin indicates pagination in the foreign text.

an automatically generated code for each of the processors of the system.

This is why dedicated languages such as HPF suggest manual mapping primitives. Based on these primitives, the programmer must himself determine good mapping. The result is that the use of the resources made available by supercomputers comprising a large number of /2 processors is far from optimum, since the mapping function is complex. Actually this function includes the division, distribution and alignment of data on various processors, the distribution of the computing and communications tasks, as well as their ordering in time. Moreover each choice relative to one of these functions is closely tied to the computer architecture and the physical characteristics of the parallel architectures.

The object of this invention is a generic process for assisting the mapping of systematic signal processing applications onto a computer with homogeneous parallel architecture, a process which allows at least one optimized mapping solution to be obtained at a level of granularity as fine as possible, based on a complete functional description of the application and of the computer used.

For each functional and physical component of the application, the process as claimed in the invention consists in establishing a model defined by a set of relations on different variables relative to this component, in order to model constraints, in concurrently resolving the relations which have been established in this way, in deducing at least one solution, and in the case in which several solutions are obtained, in choosing the one which optimizes at least one criterion.

The constraints are those relative to the subfunctions of the mapping function, i.e.: partitioning, alignment, distribution of data and sequencing of operations.

This invention will be better understood by reading the detailed description of one embodiment, taken as a nonlimiting example and illustrated by the attached drawings, the sole figure of which is a flow chart of the mapping function, implemented as claimed in the invention.

This invention relates to systematic, i.e., unconditional signal processing, not subjected to external instructions or actions. This processing is moreover deterministic and structured. This processing can be for example pulse compression or a Fourier transform (FFT) computation.

Systematic signal processing applications are formed from /3
sequences of tasks which are to be expressed by loop nest (interwoven loops with defined limits) which are well-structured and parallel. Each loop nest contains a call to a procedure or macroinstruction generally corresponding to a table transform, i.e., to a function of a signal processing library such as a FFT. This transform has been described in French patent no. 2 732 787. The processing operations are regular (not subjected to external tests) and are implemented on multidimensional signals, the data are organized in large tables with dimensions (for example source, frequency, recurrence time, pointing time) which bear vectors on which the individual processing operations are going to be implemented. The table is easily adapted to the dimensions of the sensor system, and allows a mathematical formulation of the processing operations to be given by data processing means. Thus the indices of

the variables comprising the formulas become the indices of the tables.

These procedures have a fixed execution cost, included in the specification of the application. This is globally represented by an acyclic data flow graph. The application is actually in the form of a single assignment, i.e., each element of the table is only updated once by the application. In a parallel implementation the distribution of this large table on the computation nodes changes from one processing operation to the following one, thus generating the classic problem of parallelism; the changing of an axis, or "corner turn" which consumes many communication resources.

To be able to implement the invention, the signal processing application must be functionally described, and more particularly the components of this application, i.e., the tasks.

A task, also called a routine, process or processing operation, accepts one or more data flows at the input and at the output. These flows are defined based on the input and output tables in the manner described in the aforementioned French patent no. 2 732 787, which is succinctly mentioned below. On each table under consideration, a flow represents the data accessed in reading and writing by one and only one elementary processing operation. All these data comprise an elementary access or elementary transform domain. The same processing operation /4 is reiterated on an iteration space defined by the application. Certain properties are connected to the processing node:

- **The formulation of the processing operation**: it describes the formula of the elementary transform. The output data are expressed as a function of the input data. The elementary reading and writing accesses are specified as a function of the indices of the

tables. The dimensions of the tables and the storage space required to execute a processing operation over the entire iteration space of the processing operation are likewise specified.

- **The complexity of the computation:** it expresses the computation capacity required by the processing node in operations/second. It is associated with the architecture used, and constitutes an input datum of the application.

The data flows can be conditioned by the data or the indices of the tables (it depends on the application). As for the nodes, the properties are linked to the data flows:

- **Data encoding.** They are encoded on a certain number of bits (12-16-32-64-...), they generally represent real, fixed decimal point numbers or complex numbers (encoded on two integers).
- **Flow recurrence.** This value is a function of the computing capacity of the processing operations with a flow originating from recurrences of the input flows of this same processing operation.
- Acquisition is considered a task fully comprising an output flow, and a recurrence (acquisition frequency).
- **The number of data linked to an elementary transform.** It defines the number of data required by the elementary transform of the destination node. These data are so-to-speak produced by the source node and so-to-speak consumed by the destination node. However, it happens that the data are computed and not subsequently used (all the data of a library which can be of general use are not used); this does not imply duplication of computation. /5

The elementary accesses to a table are affine functions of the indices of this table, constants and private scalar variables. The iteration space of a processing operation is for itself completely defined by the affine functions applying solely to the indices of the different tables.

More exactly, a task is broken down into two parts:

- **The external iteration space** describing the computation domain. One of the dimensions can be infinite. It constitutes time. This domain is represented by a loop nest, which is perfectly interwoven and completely parallel. There is no dependency in writing, conversely, there can be reading overlaps. This is the computation domain, which must be mapped and ordered onto the parallel computer.
- **The internal iteration space** describing all the data which are useful for computing the macroinstruction or procedure. The functions of access to the table elements are pseudo-affine functions. The modulo functions are sometimes used to take into account the cyclic nature of the sensors able to be connected to the computer.

A data flow graph can be used as a functional description formalism of the TTS (systematic signal processing) application and can originate from any classic formalism for description of the signal application provided that it contains the data specified above. In particular, it can describe the application based on the following languages:

- the language "Array-Ol" (elucidated in the aforementioned French patent 2 732 787),

- the ALPHA language,
- a language describing a set of perfectly interwoven loop nests,
- or the description formalism of the MD/SDF type.

Mapping consists in automatically distributing the signal processing operations to be carried out on a data flow, and these data themselves, onto a computer with parallel multiprocessor /6 architecture, taking into account the various constraints of the hardware resources as well as the performance imposed on the computer.

The parallel architecture involved here is a homogenous parallel architecture in which all the processors are identical and are of the SIMD/SPMD type ("Single Instruction/Program Multiple Data"), i.e., in which all the processors execute the same instruction or the same sequence of instructions (for example, a program) on different data. The routing of the data between the different processors is of the static type, that is, the data paths between the processors are dictated before initialization of each mode (they are defined during compilation of the application). At a given time the macroinstructions executed in parallel on each of the processors are identical. The data necessary for processing of the macroinstruction must be located in the local memory of the processor, which executes it.

For signal processing applications, the "dimensions" of the computer architecture used are the necessary constraints of the mapping. However these constraints are not taken into account by the classic procedures of automatic mapping (such as the procedures cited above) and thus are not treated to this end in the prior art. The characteristic parameters of said dimensions are:

- The number of available processors which are all of equal capacity.
- The capacity of a processor. For signal processing applications in real time, the latency time of the computations (time at the end of which the results of these calculations are available) is very significant. This time can be limited to a maximum value, and it depends on the capacity of the processor, which implements the computation. This capacity is expressed in the number of computation cycles per second.
- The available memory. It is uniformly distributed over all the processors of the computer.
- The characteristics of communications between processors. They are defined in terms of bandwidth, i.e., in the number of computer cycles necessary to guarantee communications of one packet of data between the processors of one pair of processors. The length of communications is then not a function of the computer topology. /7

The mapping involved here comprises the four subfunctions of partitioning, alignment, distribution and sequencing. Until now, the four subfunctions were each processed separately. Conversely, this invention calls for simultaneous processing of these subfunctions concurrently. The mapping makes it possible to find the appropriateness between a program (with parallelism which is specified or not) and a computer with homogenous parallel architecture such as indicated above. It consists is distributing the processing operations and the data on different processors of the computer and establishing their sequencing,

while optimizing the parallelism of the application.

This mapping comprises the determination of various constraints associated with the application. These constraints are on the one hand "application" limits (associated with the size of the specific elementary tasks of systematic signal processing), on the other hand the constraints connected with the architecture of the computer (number of processors, memory capacity, topology of the processor network and data rate, and finally the constraints associated with the execution (affine ordering, overlapping between data communications and computations performed).

Modelling by constraints consists essentially in establishing for each constraint a relation between at least two variables or a relation between one variable and a given value (generally a threshold). This relation is a linear relation (generally a 1st degree polynomial).

The process of the invention, proceeding from this modelling, executes concurrent (non-sequential) resolution of all the models in order to deduce therefrom one or more solutions which satisfy all the constraints. In the case of several satisfactory solutions, the one best satisfying one or more criteria can be chosen (examples of which are cited below) and it is advantageously possible to proceed heuristically.

The specification languages of the signal applications, polynomials as integers and affine applications make it possible to precisely model the mapping functionalities. Linear algebra makes it possible to synthesize different models at the level of granularity /8 required by the complexity of the general problem. These models are known in the prior art of automatic mapping [J.Li and M. Chen, "The

data alignment phase in compiling programs for distributed memory machines", Journal of Parallel and Distributed Computing, pp. 213-221, Vol. 13, 1991; P. Feautrier, "Toward automatic distribution", Parallel Processing Letters, pp. 233-244, Vol. 4, no. 3, 1994; M. Dion, "Alignment and distribution in automatic parallelization", These Informatique, ENS. Lyon, 1996] and are made the object of adaptation to the application context of the signal. They are expressed using linear and nonlinear constraints which can be integrated into the process of the invention. The process is called for on the one hand by its capacity to process the algebra specific to the different models and on the other hand by its language dimension which facilitates the expression and composition of heuristics and constraints specific to one domain, and/or complex strategies allowing control of the resolution stages. The data manipulated by the mapping system are thus the elements of the tables, the macroinstructions, of which the behavior must necessarily be modelled according to the functions, which are to be performed. All the models described below are completely formalized based on linear algebra, allowing control of the granularity for each model. Each model defines one component of the compilation of one signal processing application on a parallel computer.

- The distribution of the computations and data on the processors is in fact a problem of disjunctive resources (exclusive of one another).
- The ordering of the processing operations, taking into account the memory and communications resources represented by the capacity limits.

The objective is to optimize different criteria such as the latency of the application or the (financial) cost of the target architecture. Moreover, many models specific to the problem of mapping have been developed and follow up the description of the problem, such as for example the communications or the physical time. Taking into account the number of tasks and number of data to be considered during mapping, each model is defined in intention rather than in extension. The possibility of working at various levels of granularity is /9 fundamental for the problem of mapping, and an algebraic formulation of partitioning is used for this purpose. The latter fixes the granularity of the other models, it thus supports many relations in the very heart of the conceptual model; they are thus global constraints and the keystone of the problem to be solved. Finally, there are heuristics which are local to one or more models. But there is no global heuristic. The heart of this invention uses the multimodel approach by concurrent constraints [J. Jourdan, F. Fages, D. Rozzonelli & A. Demeure, "Data Alignment and Task Scheduling on Parallel Machines using Concurrent Model-based Programming", Proc. ILPS 94, 1994]; this allows the problem of manual mapping to be comprehensively understood.

According to the process of the invention the models are establish at the rate of one model per component, whether functional or physical. By definition a model must be regarded as a set of specifications of the behavior of the component, which it models.

The flow chart of the sole figure shows the different models implemented for the reference "mapping" function 1 in its entirety. These models are: the architecture of the target processors (2), the storage capacity (3), the partitioning of the data flows (4), the

interprocessor communications (5), the ordering of events or sequencing of computations (6), the physical time or computation time (7) and the signal inputs/outputs (8).

The different links established between these models are of two types: "hyperlinks" shown by the complex arrows (9, 10) in the form of irregular polygons, which each link several models together, and simple links represented by the lines with arrows on each of their ends and linking two models at a time.

The complex arrow **9** which corresponds to the criterion "number of processors" links the models **2**, **3**, **4**, and **5**. Complex arrow **10** which corresponds to the criterion "dependencies" links the models **3**, **4**, **5** and **6**.

Model **2** is linked by single links to models (3) (criterion /10
"memory size"), **5** (criterion "bandwidth") and **6** (criterion "programming mode").

Model **3** is linked by single links to models **4** (criterion "data volume") and **6** (criterion "distance and cardinality).

Model **4** is linked by a single link to model **7** (criterion "volume of calculation").

Model **5** is linked by single links to models **6** (criterion "communication events) and **7** (criterion "length of communication").

Model **6** is linked by a single link to model **7** (criterion "distance and cardinality).

Model **7** is linked by a simple link to model **8** (criterion "latency and recurrence").

In the models described above, the specifications of the behavior of the different components of these models are expressed on the basis

of mathematical relations. It can be deduced therefrom that the models are identified with the set of relations defined on their variables. These relations are either primitives of the language used (primitives comprising part of a library of relations), or relations defined by the user.

Due to the fact that the models themselves comprise the essence of the process, the properties of the relational paradigm (set of rules governing the relations which can be established between the models) have immediate consequences for the properties of the functional components of the process. The properties of the relational paradigm are the following:

- **Formal description:** The relations represent a formal description of the behavior of the component. Actually, for each of the subfunctions of the mapping, mathematical modelling has been formally specified. In most cases, it is the result of the work of specialists in parallelization, and it has been adapted not only to the application framework which is signal processing, but has been extended to the context of concurrent modelling.
- **"Adirectionality"**: The concept of relation makes it possible to abandon the functional paradigm based on the input/output distinction. A relation ensures complete reversibility of the arguments. This allows a distinction to made only with respect to execution, as a function of the nature of the arguments (known and unknown). /11
- **"Compositionality"**: The composition of the relational models is quite simply the logical conjunction of the relations

which comprise the model. This implies simple semantics of the compositionality. The set of solutions of a composite model is simply the intersection of the solutions of the models.

They contribute to the versatility of the program. The induced properties of the process are then:

- **An enlarged domain of use**: A model can be used in several contexts as a function of the objective to be achieved.
- **Increased interchangeability**: A model can be modified or completely redefined by the datum of a new specification without having to intervene in the other models.
- **Intrinsic compositionality**: The model of a system is synthesized on the basis of the model of its components.
- **Simple maintainability**: Maintainability remains local to each model.
- **Easy extensibility**: Extending a system comes back to composing it with the existing system.

In all known software parallelization solutions, the state of a system is characterized by the content of the memory at a given instant. Elementary operations are reading and writing on or proceeding from the memory. The state of a system is then characterized only by the set of values of storage locations linked to the variables, which comprise it. The fundamental difference between the process of the invention and the other software solutions is the representation of this memory. In the case of the invention the memory is not reduced to a set of storage locations, but in itself constitutes a constraint. The latter is able to provide partial information on the set of variables

which comprise the system. It is of interest to note that all the /12
reasoning implemented by the constraints is based on this paradigm of manipulation of partial information. The advantage of the constraints is quite simply due to the fact that the system in the course of elaboration can make decisions without having to wait for its being fully determined.

The resolution of industrial applications is not restricted to a well-defined problem, but integrates the combination of several subproblems. It is necessary to solve problems of combinatorial optimization of multicomponent, multifunction problems in which the constraints are very heterogenous and in which the different components must be considered at different levels of granularity. The invention offers solutions allowing the coexistence of partially overlapping, coordinating and decomposing models.

One alternative for the multicomponent, multifunction problems: specification of models for each component and for each function which are then combined during the resolution of an objective, makes it possible to provide one alternative to the solution of system problems. Moreover, the models are often very heterogenous. Some can be expressed solely using linear constraints, other can require symbolic or boolean constraints.

Independently of the heterogeneity of the constraints, the invention makes it possible to guarantee global coordination of the system by simple local interactions.

In the case of problems of combinatorial optimization, the process of the invention offers a good technological solution because it allows concurrent use of all redundant models during resolution.

Actually:

- The resolution of highly combinatorial problems rarely consists in a single mathematical formalization. Often, if not always, complementary formalizations are necessary. One example could be redundant formalizations which take advantage of the properties of partial solutions, another example would be consideration of the symmetries of the problems, etc.
- The problem which arises then is to use all the information at the same time. In the context of more classical approaches, such as operations research or integer programming, this stage is always very sensitive. Actually programs written in an imperative language require not inconsiderable development time and are often difficult to extend and modify. /13

The invention likewise allows solution of relational problems of the models at several levels of granularity.

The efficiency of the final implementation depends on three parameters. First of all, it depends critically on the efficiency of the system of constraints used, on the control which is exercised over the search for a solution (see below), but also largely on the granularity considered in the modelling. Experience shows that it is essential to consider different levels of granularity and to be able to reason on different levels. Here again the invention offers a solution allowing description, coexistence and coordination of models at different levels of granularity. This latter point has not yet been genuinely exploited in applications.

The "mapping" function is then modelled through different models originating from the work of specialists in "automatic parallelization" such as the following:

- **Partitioning** which expresses the distribution of computations and data on the processors.
- **The dependencies** which characterized the iterations accessing the same data.
- **Ordering** which consists in organizing the execution of parallel processing operations in time.
- **The interphase** which yields communications between two partitioned phases of computation.
- **The architecture** which is in fact a set of parameters such as the number of processors, bandwidth,...
- **Communications**.
- **The memory** which defines through one constraint of capacity for which the ability to compute an allocation is assured.
- **The real time signal** which is composed of the latency and input/output constraints (periods,...). /14

Generically, a model is composed of definitions of variables on which the constraints specific to the models are based.

The process of the invention makes it possible to simultaneously solve the problem of automatic mapping of signal processing applications on parallel machines and allows the user to manipulate the solution found without violating the constraints established by the global system. This approach is registered in the context of codesign and virtual prototyping. Based on a formal description of the application (such as for example a language of the MD/SDF type

developed at Berkeley, a functional description in ARRAY-OL or a specification using loop nests) the tool will produce a generic mapping pseudocode for the homogenous target machines under consideration. It will then be possible to directly interface this pseudocode with different compilers of the target architectures.

This process can then allow the user to do the following:

- **input a partial solution**: The user inputs a partial mapping solution and the tool pursues its search by completing the mapping solution which will be validated by conception. For example:
  - -> The user has configured his machine with an insufficient number of processors for the type of mapping he wishes. The process will make it possible to find the minimum number of processors necessary to the mapping dictated in the set of available processors.
  - -> The user can dictate the sequencing of computations. The system will then find the partitionings. i.e. the distributions of data and of computations in the memory and on suitable processors.
  - -> Likewise, the user can dictate the initial partitioning, the system will find compatible orderings.
- **play on common compromises ("trade-off")**: the process allows the user to validate the compromises between the sensitive parameters of the mapping in the application conception phase, compromises between: /15

-> number of processors/ bandwidth,

-> memory/real time,

-> number of processors/real time,

-> bandwidth/real time,

-> memory/bandwidth.

- **visualize complex solutions** such as for example ordering, division of data, assignment, etc...
- **dimension the machine**: the process allows specification of the resources necessary for mapping a particular application on a given type of machine without violating the application constraints. This consists in taking into account the dimensions and number of each hardware component.
  -> number and capacity of processors,
  -> performance of the network of interconnections,
  -> size and type of memory, synchronous/asynchronous random access memory and cache memory, hard disk),
  -> dimension of different system interfaces.
- **choose the optimization criterion of the mapping**:
  -> **machine dimension**: The process allows configuration of a minimum machine for a given application. For example, the user can choose to configure a machine with a minimum number of processors.
  -> **latency**: The process makes it possible to find the mapping(s) which minimize(s) the execution time of the application on a target machine predefined by the user.
  -> **efficiency**: The process allows maximization of the parallelism of the mapping(s) of the application on a

target machine predefined by the user.

-> **cost**: By integrating a cost (for example, financial) on each of the hardware components, the process allows the mapping(s) of the application which minimize(s) this cost to be found. /16

-> **computer operating time**: The process allows the mapping(s) of the application to be found which minimize(s) the computer operating time predefined by the user in order to be able to possibly map a second application.

-> **input or output recurrence**: The process makes it possible to find the mapping(s) which minimize(s) the input and/or output rate of the results produced by a set of processing operations. These constraints are often dictated within the framework of video, acoustic and/or very high frequency signals.

- **determine the signal parameters** as input/output modes, latency, recurrence (of input and/or output) for a given machine and application.

Moreover, the presence of the set of heuristics allows the user to lighten the burden of the search for a solution by orienting it. The engineer can in fact

- **choose one or more heuristics** in a predefined set, i.e. have the choice of different canonic mapping solutions. For example:

  -> ordering of computations to sooner or later.

- **maximize the parallelism,**

- **minimize the communications,**
- **overlap communications by computations,**
- **maximize the locality of the data in different processors.**

CLAIMS /17

1. Generic process for assisting the mapping of a signal processing application onto a computer with homogeneous parallel architecture, characterized in that for each functional and physical component of the application it consists in establishing a model defined by a set of relations on different variables relative to this component in order to model the constraints, in concurrently resolving the relations which have been established in this way, and in deducing therefrom at least one solution, and in implementing the mapping of the application.

2. Process as claimed in Claim 1, characterized in that the different models implemented are: the architecture of the target processors (2), the memory capacity (3), the partitioning of the data flows (4), the interprocessor communications (5), the sequencing of computations (6), the computation time (7) and the signal inputs/outputs (8).

3. Process as claimed in Claim 1 or 2, characterized in that in the case of obtaining several solutions, the one which optimizes at least one criterion is chosen, the choice being made heuristically.

4. Process as claimed in one of the preceding claims, characterized in that the application is described functionally by a multidimensional data flow graph.

5. Process as claimed in Claim 4, characterized in that the application is described based on one of the following languages: the

language "Array-Ol", the ALPHA language, a language describing a set of perfectly interwoven loop nests, a description formalism of the MD/SDF type.